Basic features of object oriented programming:

1. Class:

Class is an interface to create an object. Class has functions bind together. Function defines the behavior of an objects and attribute defines the property of an objects which it holds from single class any number of an object can be created.

Example:

```
class car{
  int height;
   int weight;
   string color;
   void start()
    {
cout<<"start"<<endl;
    }
 void run()
   {
 cout<<"Run"<<endl;
   }
}
```

Here, class car has attributes as weight, height, color and function as start () and run().

2. Object:

Object is a run time entity. Function and attributes comes play only after creating an object. This refers that features are defined on class where access of these features are possible only creating an object.

Example:

Car Lamborghini, ferai;

Here Lamborghini , ferai are only after real manufacturing of Lamborghini, ferari.

Note: for easy visualization, compare class with drawing of a car. By using single drawing many car can be manufactured which represents an object in c++.

3. Data hiding:

Data Hiding is the process of hiding information (data) from an unauthentic users information is provided only after the proper validation. If validation fails then information is not shown.

Example:

a) in ATM machine, people are able to access account information only after proper validation of pin number goes wrong at the time of validation then information is secure.

b) in facebook ,gmail and others social sites, password and id entered must have to match with the previously stored password and id ,resides in database to view information.

Advantages: it provides security.

4. Abstraction:

Abstraction says the detail mechanism is not necessary , just basic knowledge of using system's features is sufficient for proper utilization of system.

Example:

to use ATM machine people does not have to learn how ATM card is validated after entering into machine, which language is used to validate data, which databases either SQL or Oracle is implemented in bank to store data. Just basic knowledge of how to use ATM machine is sufficient to access account information. here detail mechanism is hidden and only basic features is provided and it's called an abstraction.

Advantages:

it provides simplicity.

5. Encapsulation:
Encapsulation is just like capsule. It combines the features of both data hiding and abstraction in single unit.
Example:
In ATM system, GUI is integrated with the database through certain programming language. If GUI is needed to be update then without customizing all the features of database and language ,it can be updated.
Advantages:
It makes customization easy.
6. Inheritance:

```
class P{                    class Q{                    class R{
250 function ()             250 function ()             250 function ()
}                               }                           }
```

Suppose,3 class are there each having 250 number of functions. Let 200 numbers of functions are common among them. we concept yet do not know about the inheritance concept yet then the total number of functions to be written =750. Say, 750 functions take 90 hour of time code.

Inheritance concept say, if common functions are there among classes then place them in one class. After that use such policy, so other classes can easily access those common functions.

```
class common{
    200 functions()
};
class P: common{
    50 functions()
};
class Q: common{
    50 functions()
};
class R: common{
50 functions()
};
```

The total number of functions to be written = 350. Total time to write 350 function takes 42 hour of time.
Advantages:
It helps to refuse code and thus reduce time to code.
7. Polymorphism:
Polymorphism refers to many forms. This means under different situation same thing can behave differently. We human being is best example of polymorphism.
In University a person is a student, in home same person can be son/daughter of parent or husband/wife or father/mother of their child. According to location, relation changes for same person.
In OOP, same concept applies in polymorphism. Function overloading and Dynamic Binding is an example of polymorphism.
Example:

```
#include<iostream>
using namespace std;
void m1(int i)
{
```

```
 cout<<"int-args:"<<endl;
 }
void m1(float f)
{
   cout<<"float-args:"<<endl;
}
int main()
{
  m1(2);
  m1(2.3f);
  return 0;
}
```

Output:

```
   int-args:
 Float-args:
```

Here, both functions have same name but depending upon type of argument it gives different output. If int argument is passes then it shows "int-args" as output. If float argument is passes then it shows "float-args" as output.

Advantages:

It provides flexibility.